

R U T C O R
R E S E A R C H
R E P O R T

BLOCKING NEGATIVE CYCLES
IN MEAN PAYOFF GAMES
IS STRONGLY POLYNOMIAL TIME
SOLVABLE

Sergei Vorobyov^a

RRR #27 - 2007, October 11, 2007

RUTCOR
Rutgers Center for
Operations Research
Rutgers University
640 Bartholomew Road
Piscataway, New Jersey
08854-8003
Telephone: 732-445-3804
Telefax: 732-445-5472
Email: rrr@rutcor.rutgers.edu
<http://rutcor.rutgers.edu/~rrr>

^aTU Wien: Vienna University of Technology, Data Bases and AI Laboratory. Email: svorobyov@gmail.com

RUTCOR RESEARCH REPORT
RRR 26-2007, OCTOBER 11, 2007

BLOCKING NEGATIVE CYCLES
IN MEAN PAYOFF GAMES
IS STRONGLY POLYNOMIAL TIME
SOLVABLE

To the memory of Peter L. Hammer (1936 – 2006)

Sergei Vorobyov

Abstract. Trying to improve the algorithm [10], Leonid Khachiyan raised the following natural combinatorial game-theoretic problem, one of the few remaining natural problems in $NP \cap coNP$ unknown to be in P, and generalizing MINIMUM COST AVERAGE CYCLES.

BLOCKING NEGATIVE CYCLES. *Given a finite edge-weighted digraph without sinks (every vertex has an outgoing edge) and a subset C of its vertices, can we choose one outgoing edge from each vertex in C (and throw away all other outgoing edges from C) in such a way that the remaining digraph has no negative-weight cycles reachable from a distinguished starting vertex?* \square

(I.e., one player forbids edges so as the other cannot construct negative cycles.) This problem appears as a subproblem in computing values and constructing ergodic partitions in the celebrated Mean Payoff Games [5, 6, 10]. Until now a strongly polynomial solution for the above problems appeared elusive. All existing algorithms were *superpolynomial*: [10, 14, 16, 2, 15] suggested *pseudopolynomial* algorithms, with the number of iterations proportional to the maximum absolute edge weight, whereas [4] described a *subexponential*, in the number of vertices, algorithm.

This paper settles the problem by showing, for the first time, that BLOCKING NEGATIVE CYCLES in cyclic Mean Payoff Games, as well as P-MEAN PARTITION (partition vertices into subsets with mean values $< p$ and $\geq p$), are *strongly polynomial time* solvable, independently of the edge weight.

Key words: cyclic games, mean payoff games, optimal strategy, strongly polynomial algorithms.

1 Introduction

Mean Payoff Games (MPG for short) were introduced and studied in [5, 13, 6, 10]. Consider a finite edge-weighted digraph $G = (V, E, w)$ with every vertex possessing an outgoing edge, the set of vertices V partitioned into subsets V_{MAX} of MAX player and V_{MIN} of MIN player, $n = |V|$, $m = |E|$, and $W = \max_{e \in E} |w(e)|$. Starting in a vertex v_0 the players move a pebble along edges *ad infinitum*, constructing a sequence of edges v_i , $i = 0, \dots, \infty$. If the pebble is in a vertex $v_i \in V_{\text{MAX}}$ then MAX selects an outgoing edge from v_i and moves the pebble to its destination vertex v_{i+1} ; otherwise MIN makes the analogous choice and move.

Players MAX and MIN are adversary, the first one wants to maximize, whereas the second one wants to minimize, respectively, the values

$$\liminf_{k \rightarrow \infty} \frac{1}{k} \sum_{i=0}^{k-1} w(v_i, v_{i+1}), \quad \text{and} \quad \limsup_{k \rightarrow \infty} \frac{1}{k} \sum_{i=0}^{k-1} w(v_i, v_{i+1}). \quad (1)$$

The *finite* zero-sum version of MPGs is similar [6]: a finite MPG develops until the first vertex repetition occurs, at which point the mean of the resulting simple cycle is declared to be the value (paid by MIN to MAX).

It turns out that the infinite and finite versions of MPGs are *equivalent* [6]. In particular, every vertex $v_0 \in V$ has a *value* $\nu(v_0)$ equal to both limits in (1). The players can secure this value, each by applying a *pure positional* strategy (i.e., a deterministic choice of one outgoing edge per his vertex) resulting in a simple cycle with the mean equal to $\nu(v_0)$. If one of the players deviates from his optimal strategy, he can only reach a worse value. Also, revealing a strategy in advance is not a disadvantage [6, 10]. Moreover, when one player fixes his pure positional strategy, an optimal counterstrategy of his adversary is polynomial time computable (e.g., by Karp's minimal average cycle algorithm). Consequently the problem whether the value of a vertex is above or below a certain threshold is in $\text{NP} \cap \text{coNP}$. [6, 10, 16]. However, until now no polynomial algorithms were known. [6, 10] suggested exponential algorithms for finding values. Later [16, 14, 2, 15] developed pseudopolynomial in W algorithms. A step forward was achieved in [4], where a subexponential in n and simultaneously pseudopolynomial in W algorithm was introduced. Finally, in the particular case of nonnegative edge weights, [12] suggested a strongly polynomial (in n) algorithm. Unfortunately, it does not extend to the most interesting general case of arbitrary edge weights.

In this paper we suggest the first *strongly polynomial* in n , m algorithm for BLOCKING NEGATIVE CYCLES and related problems. Since the possibility of blocking negative cycles

Acknowledgements: DIMACS and RUTCOR visitor, 2004–2005. Supported by the Grant IG2003-2 067 from the Swedish Foundation for International Cooperation in Research and Higher Education (STINT, www.stint.se) and the Swedish Research Council (www.vr.se) grants “*Infinite Games: Algorithms and Complexity*” and “*Interior-Point Methods for Infinite Games*”. DIMACS hosts: Leonid Khachiyan and Endre Boros.

depends on a starting vertex of the game (think of the game on two disjoint cycles, one negative and one positive), it is more convenient to consider the following, strongly polynomially equivalent, partition problem.

BLOCKING NEGATIVE CYCLES (PARTITION FORM). *Given an MPG G , find a partition of its vertices into the sets $G_{\geq 0}$, $G_{< 0}$ starting from which MAX can enforce nonnegative and MIN can enforce negative cycles.* \square

In [4] we considered the following strongly polynomially equivalent problem.

P-MEAN PARTITION: *given an MPG G and a threshold $p \in \mathbb{R}$, find a partition of its vertices into the sets $G_{\geq p}$, $G_{< p}$ starting from which MAX can enforce cycles of mean weight $\geq p$ and MIN can enforce cycles of mean weight $< p$.* \square

To address this problem, in [4] we introduced the following “controlled” analog of the standard *shortest paths* problem; cf., [11].

LONGEST SHORTEST PATHS.

Given: a weighted digraph (without 0-weight cycles) with a sink and a set of *controlled* vertices.

Find: a selection of *exactly one* outgoing edge from each controlled vertex maximizing the lengths of the shortest paths from each vertex to the sink. \square

As it was pointed out by V. Gurvich and L. Khachiyan, this problem is related to (but different from) the NP-hard problem addressed in [8]; see also [11]. The main contribution in [4, 3] was to show that the LONGEST SHORTEST PATHS problem can be solved in *randomized subexponential* (and simultaneously pseudopolynomial) time. For that purpose we developed an iterative improvement optimization algorithm based on combinatorial linear programming. BLOCKING NEGATIVE CYCLES and P-MEAN PARTITION easily reduce to LONGEST SHORTEST PATHS [4].

The following problem is from [10].

ERGODIC PARTITION: *given an MPG find the partition of its vertices into classes with equal values (and compute these values).* \square

By standard dichotomy and approximation techniques, it is (nonstrongly) polynomially reducible to 0-MEAN PARTITION, with the number of iterations proportional to the size of the game graph and $\log W$, the length of the binary representation of the largest absolute edge weight [10, 4].

Proposition 1.1 *Finding values of MPGs is polynomial time reducible to the 0-MEAN PARTITION problem.*

Proof. For an arbitrary MPG, adding a constant k to every edge weight adds k to every vertex value; multiplying every edge weight by a constant k multiplies every vertex value by k . This is because values are defined by mean values of optimal simple cycles wrt positional

strategies, and because every cycle mean changes by additive or multiplicative constant, respectively. Therefore, partitioning with a rational mean threshold reduces to the 0-MEAN PARTITION.

Values of MPG vertices are rationals with numerators and denominators up to nW and n , respectively. If a value is known to belong to an interval of length $\leq 1/n^2$, then it is uniquely determined. By dichotomizing the range $[-W, W]$ with rational thresholds, polynomially many in n and $\log W$ times, each time invoking the partition algorithm, we may uniquely determine the value of a vertex [10, 16, 4]. \square

2 Simplifying Assumptions

For technical convenience, throughout the paper, until Section 6, we assume, without loss of generality, the following special form of MPGs:

1. the game graphs are *complete bipartite*, i.e., every edge is either from V_{MAX} to V_{MIN} to or from V_{MIN} to V_{MAX} (the players strictly alternate moves), and $E = V_{\text{MAX}} \times V_{\text{MIN}} \cup V_{\text{MIN}} \times V_{\text{MAX}}$;
2. there are no 0-weight cycles.

These assumptions simplify technical details and help to avoid clutter. Readers familiar with games know that such games form a reduction class for general MPGs and may skip Section 6 altogether.

Let us note that in MPGs satisfying the assumptions above either MAX can block negative cycles from all vertices or cannot block any such cycles from any vertex, i.e., MIN can block positive cycles. Therefore, from the decision point of view such games represents a YES/NO question. Moreover, such games are *ergodic*, i.e., every vertex has the same value. By Proposition 1.1, our algorithm will be easily adaptable to answer a more general VALUE DECISION PROBLEM FOR MPGs: whether the (unique) value (of all vertices) of such a game is below or above a given threshold.

3 LP Formulation for MPGs

The definition below does not assume that an MPG is bipartite nor complete, i.e., applies to general MPGs. The idea is to say that every MAX vertex is \geq and every MIN vertex is \leq than its successors wrt weighted edges, but using equality and nonnegative *slack variables* instead of inequalities. We will later see the adequacy of this definition.

Definition 3.1 (Linear Slack Constraints [15]) *For an MPG G let S_G , called slack constraints, be the following system of linear constraints:*

1. for every edge $x \xrightarrow{w} v$ with $x \in V_{\text{MAX}}$ write constraints

$$x = v + w + s_{xv}, \quad (2)$$

$$s_{xv} \geq 0, \quad (3)$$

where s_{xv} is a MAX slack variable associated to the edge;

2. similarly, for every edge $y \xrightarrow{w'} v$ with $y \in V_{\text{MIN}}$ write constraints

$$y + s'_{yv} = v + w', \quad (4)$$

$$s'_{yv} \geq 0, \quad (5)$$

where s'_{yv} is a MIN slack variable associated to the edge. □

Unfortunately, this trick does not allow for expressing the max and min functions, which would further require that *at least one* slack associated to all outgoing edges equals zero (and would immediately solve the problem). Such a requirement is expressible as “the product of all slacks associated to outgoing edges from a vertex equals zero”. This is, however, a *nonlinear* constraint, leading to a *nonconvex* and *nonconcave* program. We will achieve the same goal differently, in a more sophisticated way.

We adopt the convention that primed s' and w' denote MIN slacks and weights of edges outgoing from MIN vertices, while unprimed s and w will denote slacks and weights associated with MAX vertices. MAX variables will be denoted x and MIN variables by y . In the sequel we will freely identify edges with their corresponding equality constraints and vertices with their corresponding variables.

The simple LP-formulation above is adequate for expressing interesting MPG properties to be discussed below. We start with the simplest, but very useful

Proposition 3.2 *For a cycle in an MPG G let w_i, s_i, s'_i (for $i \in I$) be the weights of all edges, all MAX slacks, and all MIN slacks on the cycle. Then S_G implies*

$$\sum_{i \in I} w_i + \sum_{i \in I} s_i - \sum_{i \in I} s'_i = 0. \quad (6)$$

Proof. Just sum up left- and right-hand sides of the equalities corresponding to edges on the cycle. Variables x_i, y_j disappear by telescoping. □

This proposition partially explains why the bipartite assumption is useful. Indeed, whenever a positive weight cycle traverses only MAX vertices in G , or a negative weight cycle traverses only MIN vertices, the system S_G is infeasible, because (6) cannot be satisfied.

4 The Target Function

We will be interested in minimizing, for the reasons to become clear shortly, the following linear target function:

$$f(x, y, s, s') = 1^T s + n^2 \cdot 1^T x, \quad (7)$$

where n is the total number of game vertices, and x, y, s, s' are vectors of MAX and MIN variables and their associated slacks. Two occurrences of 1 denote the all-ones vectors of appropriate dimensions. As usual, vectors are columns, the superscript T denotes the matrix (vector) transposition, and the scalar product is denoted by juxtaposition.

Thus, the target function (7) is just the sum of all MAX slacks plus the sum of all MAX variables times n^2 (a constant for every fixed dimension n). The purpose of this multiplication will become clear later in the proof of Proposition 5.8.

5 Properties of Feasible Solutions and Target Function

Recall that we globally assume preconditions stipulated in Section 2, thus omitting them from the premises of all claims.

Feasible solutions of MPG LPs can be shifted up and down:

Proposition 5.1 *If x, y, s, s' is a feasible solution to the linear constraints S corresponding to an MPG, then $x + \delta 1, y + \delta 1, s, s'$ is also a feasible solution to S , where $\delta \in \mathbb{R}$ and two occurrences of 1 are all-ones vectors of appropriate dimensions.*

Proof. Immediate, due to bipartiteness. □

We will call the transformation from the above Proposition the δ -shift of a feasible solution.

As a consequence, to work with finite optima, we can (have to) arbitrarily fix the value of one variable as follows.

Proposition 5.2 *Let S be the linear constraints corresponding to an MPG. Then the optimum of the LP*

$$\text{minimize } f(x, y, s, s') \text{ subject to } S \cup \{y_1 = 0\} \quad (8)$$

is finite.

Proof. Every feasible solution should satisfy $x_i \geq y_1 + w_{i1} = 0 + w_{i1}$, and $s \geq 0$. □

Note that this is not true, if complete bipartiteness is not required.

We could have selected to fix any other MIN variable $y_i = 0$ instead. Let us denote by LP_i the linear program (8) with the constraint $y_i = 0$ instead of $y_1 = 0$, which will be useful in the future development.

The next property is crucial. Call an edge with associated slack 0 a *tight edge*.

Proposition 5.3 *Every minimal solution of (8) satisfies the following 0-in-out property:*

for every vertex at least one incoming or outgoing edge is tight.

Proof. Suppose a MAX-variable has all associated slacks, corresponding to incoming and outgoing edges, greater than 0. Then the value of this vertex as well as the values of all these slacks can be simultaneously diminished by some value δ (e.g., by the minimum of all such slacks), keeping feasibility and decreasing the value of the target function (7).

Similarly, if a MIN-variable has all associated slacks nonzero, its value can be *increased*¹ by a positive (e.g., by the minimum of these slacks δ) value, and all associated slacks decreased by the same value. This gives a feasible solution with a smaller target value. (A minor modification is needed for the distinguished variable y_1 : we first decrease ALL variables by δ , then proceed as before.) \square

0-in-out property was introduced and investigated in [15].

For brevity, feasible solutions satisfying the 0-in-out property will be called *0-in-out feasible solutions*. Some such solutions are very important for our purposes, since they immediately represent solutions to the games (which once again illustrates the adequacy of the LP-formulation):

Proposition 5.4 *For a feasible solution of the system of slack constraints S corresponding to an MPG the following holds:*

1. *if every MAX-vertex has an outgoing tight edge, then MAX can enforce positive weight cycles from every vertex;*
2. *if every MIN-vertex has an outgoing tight edge, then MIN can enforce negative weight cycles from every vertex.*

(Note that at most one² can hold because of our assumptions and determinacy of MPGs.)

Proof. By Proposition 3.2, the sum of slacks and weights along any cycle that results is

$$\sum_{cycle} s_i - \sum_{cycle} s'_j + \sum_{cycle} w_k = 0. \quad (9)$$

¹This explains why we do not include MIN variables in the target function.

²Below we show that there is a feasible solution satisfying exactly one of the alternatives.

Since the first term is 0 whenever MAX uses only his tight edges, the third term is positive; recall we assume wlog the absence of 0-weight cycles. In case 2, MIN always uses his tight edges and analogously, by (9), secures negative cycles. \square

Not every feasible solution with the 0-in-out property satisfies the cases 1 or 2 of Proposition 5.4. What happens in general for such a solution? Here is the answer.

Definition 5.5 (NX-Partition) *Consider an arbitrary 0-in-out feasible solution to the system of linear constraints corresponding to an MPG. Everything below will be defined wrt this solution. Let N_0 be the set of MIN-vertices without outgoing tight edges, and X_0 be the set of MAX-vertices without outgoing tight edges.*

(Temporarily) Ignore all non-tight edges. Let N be the set of all vertices from which MAX can force the play into N_0 , and X be the set of all edges from which MIN can force the play into X_0 , if both players use only tight edges. \square

We may assume wlog that both N_0 and X_0 are nonempty; otherwise, Proposition 5.4 applies immediately. Additionally, both N and X are *acyclic*, by the absence of 0-weight cycles.

We have the following properties justifying the term “partition” in Definition 5.5.

Proposition 5.6 1. *The sets N , X form the partition of the game vertices.*

2. *There are no tight edges of MAX from X to N and no tight edges of MIN from N to X . Although, there may be tight edges of MAX from N to X and tightly edges of MIN from X to N ; see Figure 1.*

Proof. Since N , X are acyclic, apply topological sorting. Proceeding in the resulting topological ordering, start constructing X , N by setting $X = X_0$, $N = N_0$. Then continue as follows:

1. if all tight successors of a MAX-vertex v are already determined to be in X , then $v \in X$ as well;
2. if some tight successor of a MAX-vertex v is in N , then $v \in N$ as well;
3. if at least one tight successor of a MIN-vertex v is in X then $v \in X$ as well;
4. if all tight successors of a MIN-vertex v are already determined to be in N , then $v \in N$ as well.

Since we proceed in the (acyclic) topological ordering, for every vertex its X or N its membership is always uniquely defined; thus X , N form a partition.

The second claim follows directly by definition of N , X and by inspection. \square

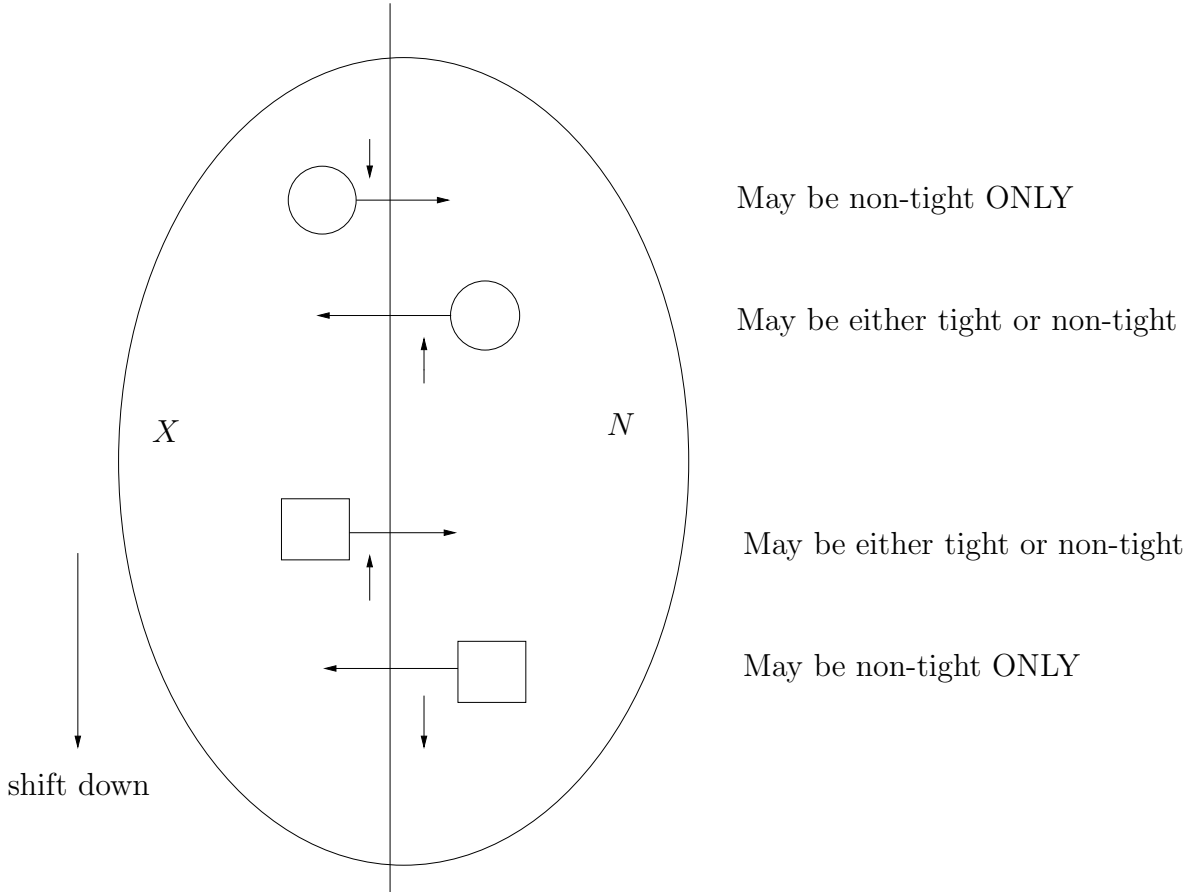


Figure 1: A nontrivial NX -partition with possible edges crossing the cut. Round (square) vertices and outgoing edges belong to MAX (MIN, resp.). Vertical arrows near the edges show what happens with associated slacks when the whole X -partition is drawn down, which is always possible since the corresponding slacks are strictly positive.

A partition is called *nontrivial* if both X and N are nonempty. Figure 1 graphically represents such a partition. The following proposition shows how one 0-in-out feasible solution determines another one by “shifting down” (or decreasing) simultaneously all vertices in X .

Proposition 5.7 *For every 0-in-out feasible solution to the system of linear constraints S for an MPG defining a nontrivial NX -partition, one can define another 0-in-out-feasible solution by shifting down the X -part as follows:*

1. *select the minimal slack δ among associated to MAX edges from X to N and to MIN edges from N to X (recall that all such edges are non-tight, so $\delta > 0$);*
2. *decrease all values of vertices in X by δ .*

Proof. Obviously, the result of the modification will again be a 0-in-out feasible solution.

This is because of the simple invariant that holds during the shifts: if a vertex has an outgoing tight edge, it will still have one after the shift. \square

We postpone the discussion of the minor problem when the fixed variable $y_1 = 0$ appears to be in X . This can be easily repaired by first shifting up all x, y variables by δ .

Note that after the shift down, an NX -partition may change nonmonotonically³, and has to be recomputed, because new tight edges appear. Such iterative recomputation always terminates and leads to the final *trivial* NX -partition (one of X or N is empty), when Proposition 5.4 applies. The associated algorithm (very similar to the potential transformation algorithm of [10]) is described in [15]. Unfortunately, this algorithm can make too many iterations of drawing-down and recomputing changing NX -partitions. Actually, it is *pseudopolynomial*, i.e., $O(W \cdot \text{poly}(n))$, where W is the maximal absolute edge weight in a game. There are known examples of games on which the algorithm actually makes $O(W)$ iterations.

Our new algorithm described here is still based on Proposition 5.7, but we do not proceed iteratively. Instead, we make *just one step*, relying on the following simple but crucial

Proposition 5.8 *During the shift down of the partition X as described in Proposition 5.7 the value of the target function (7) decreases.*

Proof. Let n_{MAX} and n_{MIN} be the numbers of MAX and MIN vertices in the game, p be the number of MAX vertices in X , q be the number of MIN vertices in X , and $n = n_{\text{MAX}} + n_{\text{MIN}}$.

Let us count (see Figure 2):

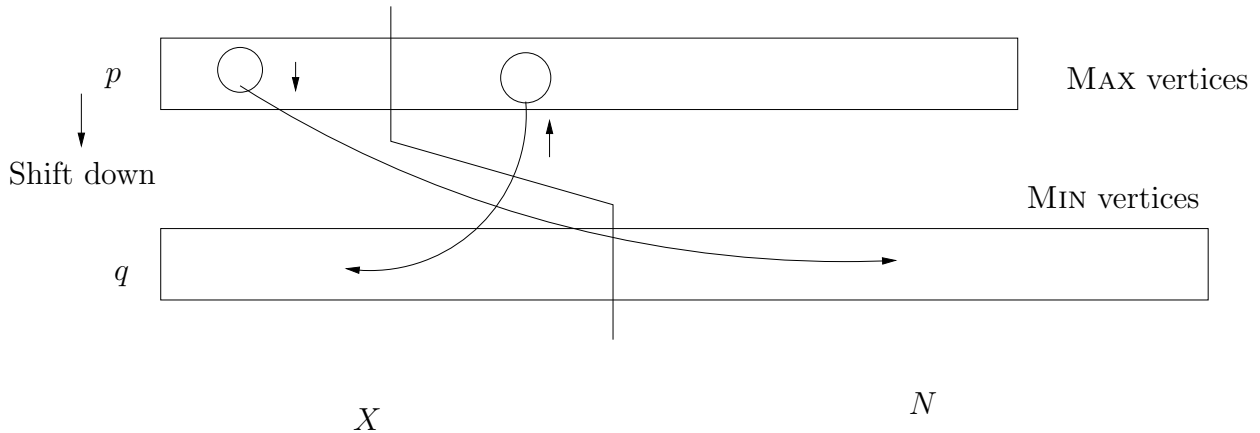


Figure 2: An X shift down. Vertical arrows show slack decrease/increase.

1. the number of MAX edges from X to N – these *decrease* their associated slacks (see Figure 2) – equals $p \cdot (n_{\text{MIN}} - q)$;

³i.e., in a sequence of shifts the sets N, X may grow and shrink.

2. the number of MAX edges from N to X – these *increase* their associated slacks (see Figure 2) – equals $(n_{\text{MAX}} - p) \cdot q$.

Therefore, the overall decrease of the sum of MAX slacks per unit shift down⁴ equals

$$p \cdot (n_{\text{MIN}} - q) - (n_{\text{MAX}} - p) \cdot q, \quad (10)$$

which can be either positive or negative. The worst case is represented by $p = 1$, $q = n_{\text{MIN}} - 1$, when the X shift down has the maximal *increase* (rather than decrease) in the sum of MAX slacks appearing in the target function. This increase is bounded from above by $(n - 1)^2$.

However, in every unit shift down of X the value of the target function (7) decreases by at least n^2 due to its second term, since at least one MAX vertex in X decreases. Hence, the overall target value decreases by at least one per every unit shift down of the X partition. \square

We are about to make the final step.

Proposition 5.9 *An optimal solution to (8) satisfies the 0-in-out property and defines a trivial NX -partition (thus solving the game by Proposition 5.4).*

Proof. The first claim is already proved as Proposition 5.3.

For the second claim, assume, toward a contradiction, that a minimal solution to (8) defines a *nontrivial* NX -partition. If $y_1 \in N$, then Proposition 5.8 allows for a feasible solution with a smaller target value by shifting down X , which gives an immediate desired contradiction.

In the case $y_1 \in X$ the argument is slightly more difficult, since X cannot be immediately shifted down keeping $y_1 = 0$ fixed.

Note that feasible domains of all LP_i (i.e., (8) with $y_1 = 0$ replaced by $y_i = 0$) are in 1-1-correspondence and differ by a shift: (x, y, s, s') is feasible for LP_i iff $(x + \delta, y + \delta, s, s')$ is feasible for LP_j for $\delta = -y_j$, where $x + \delta$ denotes the addition of δ to every components of vector x . Thus, optima for LP_i and LP_j are also in 1-1-correspondence and differ by a constant $n^2 n_{\text{MAX}} \delta$. Now, suppose, a 0-in-out feasible solution (x, y, s, s') for LP_i defines a nontrivial NX -partition and $y_i \in X$, i.e., cannot be immediately decreased by the above shift construction. Since $N \neq \emptyset$, some $y_j \in N$, and the solution $(x + \delta, y + \delta, s, s')$ of LP_j can be decreased by shifting down its X part by the above shift construction, since $y_j \notin X$. Hence, $(x + \delta, y + \delta, s, s')$ is not optimal for LP_j , thus (x, y, s, s') is not optimal for LP_i , i.e., there is a feasible solution with a smaller target value, which is enough for proving our claim by contradiction. \square

Now we have everything ready for solving MPGs satisfying assumptions of Section 2 in strongly polynomial time.

Theorem 5.10 BLOCKING NEGATIVE CYCLES *can be solved in strongly polynomial time.*

⁴We may assume wlog that the solutions to LPs and shifts are integral.

Proof. By Proposition 5.9, it suffices to write linear slack constraints (Definition 3.1) for an MPG and solve the resulting LP (8) (Just one LP is enough: by the last paragraph of the previous proof, optima of different LP_{*i*}'s differ by a non-essential constant shift only.) By Proposition 5.4, an optimal solution determines an optimal strategy for the winner, who enforces either positive or negative cycles from all vertices.

Note that the matrix of the LP (8) is a sparse $m \times (n + m)$ (with $m = 2n_{\text{MAX}} \times n_{\text{MIN}}$ constraints) matrix with exactly 3 nonzero ± 1 coefficients per row.

It is well known [9, Section 6.6] that such LPs can be solved in *strongly* polynomial time, depending on the dimension of the constraint matrix, i.e., on n, m only, *independently* of the right-hand sides (weights of edges, which can be arbitrarily large) and the target function coefficients (which do not represent a problem in our case; the size of the target function (7) is polynomial in n, m). \square

Corollary 5.11 P-MEAN PARTITION *can be solved in strongly polynomial time.* \square

6 Reduction to MPGs on Complete Bipartite Graphs

For completeness, we prove the following

Proposition 6.1 *In the 0-MEAN PARTITION problem the following assumptions can be done without loss of generality:*

1. *the game graph has no 0-weight cycles,*
2. *the game graph is bipartite,*
3. *the values of all vertices are of the same sign;*
4. *the game graph is complete bipartite.*

Every reduction from general MPGs to the restricted case 1, 2, 3, and 4 is polynomial.

Proof. Given an arbitrary MPG, consider the following chain of reductions.

1. Multiplying every edge weight by $n + 1$ and adding 1 does not change signs of positive- and negative-weight cycles, but 0-weight cycles (if any) become positive-weight. The 0-mean partition remains the same.

2. The straightforward solution is to introduce a vertex of the opposite player between two vertices of the same player. This, however, may increase the number of vertices quadratically. A more economic solution, leading to just a linear increase in the number of vertices is depicted in Figure 6 (the outgoing edge from the new vertex gets weight 0). Note that this transformation may actually change means of cycles, but not 0-means partitions, which is enough for our purpose.

3. Let v be an arbitrary vertex of a bipartite MPG G without 0-weight cycles. Construct G' by adding a new backward edge from every vertex $u \neq v$ of G to v of weight $-M$ for an

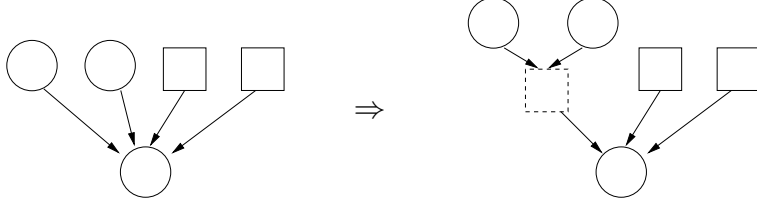


Figure 3: The transformation to a bipartite game for a MAX node (round), with only a linear increase in the number of nodes.

edge from a MAX vertex and of weight $+M$ for a MIN vertex, where $M = (n - 1)W + 1$. Suppose, a play in G' starts from v . If MAX can secure a positive value of v in G , he can use the same strategy as in G never using new edges. If MIN never uses his new edges, then the value is the same as in G . But if MIN is the first to use his heavy edge back to v , the cycle thus formed has a mean $\geq [(n - 1)W + 1 - (n - 1)W]/n > 0$ (and we refer to the equivalence of finite and infinite MPGs [6]). The case when v has negative value in G is symmetric.

Now suppose a play starts in any other vertex $v' \neq v$. Then each player can reach v and then follow the same strategy as he uses from v . The signs of means in *infinite* plays thus formed, one starting from v' , the other from v , are the same (the initial finite path does not matter (this argument also depends on the equivalence between finite and infinite MPGs [6])). Hence, in G' mean values of all vertices are of the same sign.

4. Let $M = (n - 1)W + 1$. Add all missing edges between V_{MAX} and V_{MIN} with the weight $-M$ or $+M$ depending on whether an edge leaves a MAX or a MIN vertex. This makes the graph complete bipartite and preserves the signs of all values. The choice of M is motivated in the same way as in 3. Note that we can also assume both partitions have the same number of vertices, by duplicating vertices and their outgoing edges. \square

Remark. In the chain of reductions above the number of vertices and edges may grow just linearly in the number of vertices n . In contrast, maximum absolute weights in 1, 3, and 4 are multiplied by n each time, which results in the overall weight multiplication by n^3 . Our algorithm operates on bipartite MPGs without 0-weight cycles. Thus, assumptions 1, 2 cost us a factor of n in the weight increase. \square

7 Conclusions

By Propositions 1.1 and 6.1 we get

Corollary 7.1 ERGODIC PARTITION FOR MPGs *is solvable in pseudopolynomial time* $O(\text{poly}(m, n) \cdot \log W)$. \square

Note, for comparison, that [10, 16, 2] all suggested pseudopolynomial, but $O(\text{poly}(n, m) \cdot W)$ algorithms, whereas the algorithm in [4] has simultaneously pseudopolynomial and subexponential bounds: $\min(O(\text{poly}(n, m) \cdot W), e^{O(\sqrt{n \log m})} \cdot \log W)$, and the algorithm [1] improves the above bound by dropping the last $\log w$.

Similarly, we have

Corollary 7.2 *The decision version of the LONGEST SHORTEST PATH problem (whether the longest shortest path from a vertex to the sink is longer than a given threshold?) is strongly polynomial time solvable.* \square

The previous best algorithm [4] has bound $\min(O(\text{poly}(n, m) \cdot W), e^{O(\sqrt{n \log m})})$.

In conclusion, let us mention an application. *Parity Games* are similar to MPGs, but instead of weighted edges have colored vertices, assigned natural priorities. Numerous temporal logics of programs and model-checking problems reduce to Parity Games [7]. Similarly to MPGs, a parity game starts in a vertex, and players MAX and MIN construct an infinite sequence of vertices. MAX tries to ensure that in every such sequence the largest vertex color appearing *infinitely often* is *even*, whereas MIN tries to make it *odd*. The (folklore) reduction from Parity Games to MPGs consists in assigning all edges from a vertex of color c the same weights $(-n)^c$. It easily follows, since a cycle is positive iff the largest color on the cycle is even, that Parity Games are determined in pure positional strategies and solvable in strongly polynomial time, by reduction to a single instance of BLOCKING NEGATIVE CYCLES or 0-MEAN PARTITION. Thus,

Corollary 7.3 *Parity Games are strongly polynomial time solvable.* \square

In this paper we left out the question of precisely determining the best possible polynomials in the strongly polynomial or pseudopolynomial bounds. These depend on the best available strongly polynomial algorithms for linear programs of the special form (see the proof of Theorem 5.10). The standard strongly polynomial refinements of LP are based on Khachiyan's ellipsoid algorithm [9], with quite (prohibitively) high polynomial degrees. The very special structure of LPs resulting from MPGs allows for expecting better polynomials in this restricted case. This requires further investigation similar to the two-variable-per-constraint subclasses of LP. After all, game theorists have to contribute better game-specific algorithms compared with the general LP algorithms. Another interesting question is the generalization of the standard MPGs considered here to extended MPGs with the node-wise edge interdictions [14, 12, 11]. We conjecture that the results of these papers can be improved and generalized, respectively. The remaining open problem is strong polynomial solvability of the ERGODIC PARTITION problem for MPGs. Although computing values requires $O(n \log W)$ time just to print out the answer in binary, it is conceivable that the partition into ergodic classes can be done more efficiently, by a strongly polynomial (in n, m) algorithm. See [1] for a strongly subexponential one.

References

- [1] D. Andersson and S. Vorobyov. Fast algorithms for monotonic discounted linear programs with two variables per inequality. Preprint NI06019-LAA, Isaac Newton Institute for Mathematical Sciences, Cambridge, United Kingdom, May 2006.

- [2] H. Björklund, O. Svensson, and S. Vorobyov. Linear complementarity algorithms for mean payoff games. Technical Report DIMACS-2005-05, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, February 2005. <http://dimacs.rutgers.edu/TechnicalReports/>.
- [3] H. Björklund and S. Vorobyov. Combinatorial structure and randomized subexponential algorithms for infinite games. *Theoretical Computer Science*, 349(3):347–360, 2005.
- [4] H. Björklund and S. Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. *Discrete Applied Mathematics*, 155(2):210–219, 2007.
- [5] A. Ehrenfeucht and J. Mycielski. Positional games over a graph. *Notices Amer. Math Soc.*, 20:A–334, 1973.
- [6] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journ. of Game Theory*, 8:109–113, 1979.
- [7] E. A. Emerson. Model checking and the Mu-calculus. In N. Immerman and Ph. G. Kolaitis, editors, *DIMACS Series in Discrete Mathematics*, volume 31, pages 185–214, 1997.
- [8] D. R. Fulkerson and G. C. Harding. Maximizing the minimum source-sink path subject to a budget constraint. *Mathematical Programming*, 13:116–118, 1977.
- [9] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 2 edition, 1988.
- [10] V. A. Gurvich, A. V. Karzanov, and L. G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 28(5):85–91, 1988.
- [11] L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao. On short path interdiction problems: total and node-wise limited interdiction. RUTCOR Research Report RRR 25-2006, RUTCOR, Rutgers Center of Operations Research, October 2006.
- [12] L. Khachiyan, V. Gurvich, and J. Zhao. Extending Dijkstra’s algorithm to maximize the shortest path by node-wise limited arc interdiction. RUTCOR Research Report RRR 31-2005, RUTCOR, Rutgers Center of Operations Research, October 2005.
- [13] H. Moulin. Extensions of two person zero sum games. *J. Math. Analysis and Applic.*, 55:490–508, 1976.
- [14] N. Pisaruk. Mean cost cyclical games. *Mathematics of Operations Research*, 24(4):817–828, 1999.

- [15] O. Svensson and S. Vorobyov. Linear programming polytope and algorithm for mean payoff games. In *Second International Conference on Algorithmic Aspects in Information and Management*, volume 4041 of *Lecture Notes in Computer Science*, pages 64–78, Hong Kong, China, 20-22 June 2006. Springer.
- [16] U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.